(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property
Organization
International Bureau

(43) International Publication Date
7 October 2004 (07.10.2004)

PCT

(10) International Publication Number
WO 2004/086248 A1

(54) Title: SYSTEMS AND METHOD FOR OPTIMIZING TAG BASED PROTOCOL STREAM PARSING
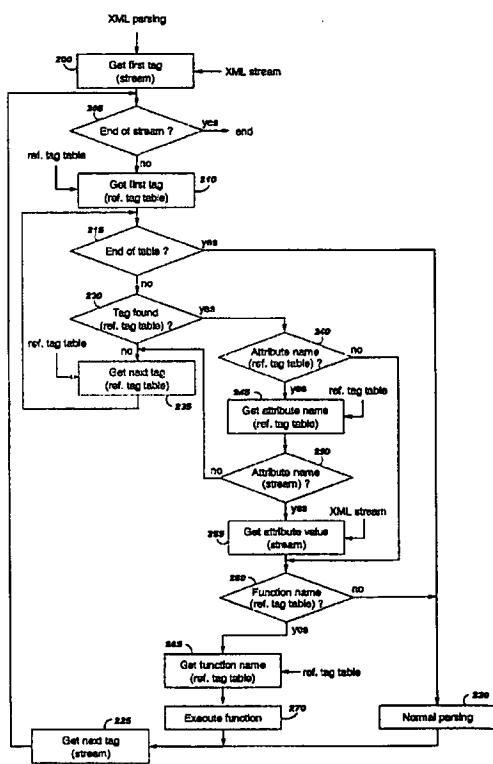
(57) Abstract: A method to optimize tag based protocol parsing
adapted to execute functions according to tags of a tag based
protocol stream is disclosed. A reference tag table, associated
to the document type definition of the tag based protocol stream,
comprises tags and corresponding attribute and function names.
When the optimized tag based protocol parser gets a tag from
the tag based protocol stream, it determines whether or not the
tag is present in the reference tag table. If the tag is not found in
this table, normal parsing continues. Else, if the tag is found in
this table and if there is at least one associated the attribute names
from the reference tag table and from thè tag based protocol
stream. If the attribute names match, the parser gets the attribute
values from the tag based protocol stream before getting the
name of the function associated to the tag and attribute names
combination. Then, the parser executes the function.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HU, IE, IT, LU, MC, NL, PT, RO, SE, SI, SK, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Declaration under Rule 4.17:
— *of inventorship (Rule 4.17(iv)) for US only*

Published:
— *with international search report*

— *before the expiration of the time limit for amending the claims and to be republished in the event of receipt of amendments*

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

1

SYSTEMS AND METHOD FOR OPTIMIZING
TAG BASED PROTOCOL STREAM PARSING

Copyright notice

A portion of the disclosure of this patent document
5   contains material which is subject to copyright protection.
The copyright owner has no objection to the facsimile repro-
duction by anyone of the patent disclosure, as it appears in
the United States Patent and Trademark Office patent file or
records, but otherwise reserves all copyright rights
10 whatsoever.

Field of the Invention

The present invention relates generally to the encoding
and decoding of electronic documents and more specifically
to systems and method for optimizing the parsing of tag
15 based protocol streams.

Background of the Invention

Recently, communications between computer systems for
data and information exchange has been significantly devel-
oping thanks to the Internet, which is known to have rapidly
20 widespread on a global level by virtue of being supported by
public communication networks, both traditional and techno-
logically advanced ones, such as the ISDN, the ADSL, the
GPRS, and others.

Success of this phenomenon is indeed due, also, to the
25 availability, in real time and cheaply, of information and

2

data stored on servers located all over the globe and connected through dedicated digital lines to computers reachable through the various last mile network access services.

5   Most of the electronic texts available from the World Wide Web are formatted according to the Hyper Text Markup Language (HTML) standard. Unlike other electronic texts, HTML 'source' documents, from which content text is displayed, contain embedded textual tags. HTML is designed
10   to display the data and to focus on how data looks. However, since HTML presents several drawbacks, in particular the inevitable pre-definition of tags, eXtensible Markup Language (XML) has been created by the World Wide Web Consortium (W3C). XML is designed to describe data and to
15   focus on what data is. Like HTML, XML is based on Standard Generalized Markup Language (SGML). Although SGML has been used in the publishing industry for decades, its perceived complexity intimidates many people that otherwise might have used it.

20   Using XML, a meaning may be assigned to each tag of the document so that it is easy for a machine to process the information. For example, a postal code may be easily extracted from a document by simply locating the content surrounded by special tags that could be *<postal-code>* and
25   *</postal-code>*, technically referred to as the *<postal-code>* element.

        There are three common terms used to describe parts of an XML document, *tags*, *elements*, and *attributes* :

        •   a tag is a string of characters comprised between the
30   left angle bracket '<' and the right angle bracket '>'.

There are starting tags and ending tags, an ending tag corresponding to the starting tag wherein a '/' is inserted between the left angle bracket and the text.

•   an element corresponds to the starting tag, the ending
5   tag and everything in between.

•   an attribute is a name and/or an associated value included inside a tag of an element.

     For example, considering the following XML document,


```
<address>
10     <name>
         <title>Mrs.</title>
         <first-name>
             Mary
         </first-name>
15       <last-name>
             McGoon
         </last-name>
       </name>
       <street>
20         1401 Main Street
       </street>
       <city state='NC'>AnyTown</city>
<postal-code>
34829
25 </postal-code>
</address>
```


     tags <name> and </name> represent a starting tag and an ending tag, respectively, the <name> element contains three child elements <title>, <first-name> and <last-name> and
30   state is an attribute of the <city> element.

4

Since XML is designed to describe data, it simplifies data interchange and enables smart code wherein important information may be easily identified.

5 US Patent n°6,480,865 describes a method for annoting XML documents with dynamic functionality. The dynamic functionality comprises invocations of Java objects. These annotations belong to a different name space, and thus a Dynamic XML-Java (DXMLJ) processor recognizes elements within the XML document that are tagged with DXMLJ prefix
10 tags, processes each of these tags, and transforms the XML document accordingly.

For handling XML documents, software applications are generally using an XML parser that may be considered as an interface between the document and the software. In such
15 case, the XML parser extracts data from the XML documents to build its internal tree representation so as to provide the software applications with the required data. A parser used with XML document is qualified as a Document Object Model (DOM) parser or Simple API for XML (SAX) parser.

20 DOM parser creates a DOM tree in memory for an XML document. A DOM parser is usually used to manipulate the document, traverse it back and forth. However, since the main drawback of DOM parser is to be memory consuming, it is reserved to handle small documents.

25 SAX parser is an event-based driven interface and invokes callback methods when a tag is encountered. It is mainly used when no structural modification is planned and can handle huge documents.

5

These parsers may be validating or non validating
parser. A validating parser checks the XML file against the
rule imposed by the Document Type Definition (DTD) while a
non validating parser does not validate the XML file against
5   a DTD. Both validating and non validating parsers check for
the well formedness of the XML document. A DTD specifies
constraints on the valid tags sequences that can be in the
document.

Therefore, since the use of XML document type is
10 increasing dramatically over the Internet, as are the
occurences of large data objects in the contents to be
transmitted, there is a need for optimizing the parsing of
such document so as to improve the processing time of
software applications handing this electronic document
15 format.

6

Summary of the Invention

Thus, it is a broad object of the invention to remedy the shortcomings of the prior art as described here above.

It is another object of the invention to provide a
5   method for optimizing tag based protocol parsers.

It is a further object of the invention to provide optimized tag based protocol parsing adapted for handling a reference tag table associating functions to tags.

It is still a further object of the invention to
10 provide optimized tag based protocol parsing, executing functions in accordance to tags of a tag based protocol stream.

The accomplishment of these and other related objects is achieved by a method for optimizing tag based protocol
15 stream parsing, using a reference tag table comprising at least one tag and a corresponding function name, said method comprising, each time a tag is read from said tag based protocol stream, the steps of:

- comparing said read tag and the tags of said reference
20 tag table and,

- if said read tag belongs to said reference tag table, determining if a function name is associated to said tag belonging to said reference tag table and,

- if a function name is associated to said tag belonging
25 to said reference tag table, executing the function corresponding to said function name associated to said tag belonging to said reference tag table.

7

Further advantages of the present invention will become apparent to the ones skilled in the art upon examination of the drawings and detailed description. It is intended that any additional advantages be incorporated herein.

5   **Brief Description of the Drawings**

Figure 1   depicts an example of a reference tag table illustrating the association of attribute and function names to a tag.

Figure 2   illustrates schematically an example of an optimized XML parser algorithm according to the invention.

Figure 3   shows an example of XML document content to illustrate the use of the invention.

Detailed Description of the Preferred Embodiment

According to the invention a reference tag table is provided, containing attributes that can enhance tags of tag
10   based protocols, with the purpose to optimize the parsing task. Such reference tag table is associated to the tag based protocol document or to another table associated to the tag based protocol document. For example, all the XML streams and documents comprise an indication of the Document
15   Type Definition (DTD) location, including the name of the DTD e.g., name.dtd, so that the XML parser may validate the XML stream or document. Using this name and/or this location indication, the XML parser may also determine the reference tag table to be used i.e., the reference tag table associ-
20   ated to the DTD e.g., name.rtt. Naturally, to handle the reference tag table, the parser needs to implement the

8

necessary logic to read the table, retrieve the possible attributes, and execute the corresponding function.

Figure 1 illustrates an example of a reference tag table 100. As Shown, reference tag table 100 comprises three 5   columns 105-1 to 105-3 and n rows 110-1 to 110-n. The first column i.e., column 105-1, is used to store the name of the tags, the second column i.e., column 105-2, memorizes the attributes corresponding to the tag of column 105-1 and the last column i.e., column 105-3, memorizes the name of the 10   function associated to the tag of column 105-1. For example, the tag *image* stored in column 105-1 and row 110-1 is associated to the attributes *save* and *skip* and to the function *SAVE_SKIP*, belonging to the same row 110-1, in columns 105-2 and 105-3, respectively.

15   According to the invention, each time the parser detects a tag, a lookup is done in the reference tag table. If the tag corresponds to one of the tag memorized in the reference tag table, the parser determines if one or several attributes are specified in the table for the tag being 20   processed and extracts the name of the function associated to the tag. If there is no function associated to the tag, normal parsing continues. If there are one or several attributes associated to the tag, the parser determine the corresponding values from the document or stream. The 25   attributes are passed as parameters to the function speci-fied for the tag and the function is executed. If there is no attribute associated to the tag, the function is directly executed. If the tag does not correspond to one of the tag memorized in the reference tag table, normal parsing 30   continues.

9

     As shown on figure 1, a plurality of attributes may be assigned to a single tag. Depending upon the parser and the nature of the attributes, several cases could be encountered. If all the attributes of a tag are required to

5   execute the associated function, the reference tag table comprises only one row for such tag. In such case, the function is executed only if, in the document or stream, a value is provided for each attribute of the tag. If some of the attributes of a tag are optional, it may be handled

10   either by the reference tag table or by the parser itself. If it is handled by the reference tag table, the same tag is repeated in several rows, each row being dedicated to a particular attribute combination (there are as many rows as there are possible combinations). Else, if it is handled by

15   the parser, a single row is dedicated to each tag and the parser analyzes the document or stream to extract the attribute values so as to determine the used optional attributes.

     For sake of illustration, the system of the detailed

20   description uses a reference tag table comprising, for each tag, as many rows as the number of possible combinations. Furthermore, in this detailed example, the table must be ordered in such a way that in the cases of multiple occurrences of the same tag in the table, the occurrences which

25   carry the highest number of attributes come first in the table.

     Figure 2 illustrates schematically an example of the algorithm of the logic to be added to an XML parser, according to the invention. After having read the first tag from

30   an XML stream (box 200), a test is done to determine whether or not the end of the stream has been reached (box 205). If the end of the stream is detected, the process stops. Else

10

if the end of the stream is not detected, the first tag of
the reference tag table associated to the DTD used for
processing the XML stream i.e., the reference tag table
associated to the XML document, is looked up (box 210) and a
5   test is performed to determine whether or not the end of the
reference tag table has been reached or not (box (215). If
the end of the reference tag table has been reached, normal
parsing continues (box 220), the next tag of the XML stream
is looked up (box 225) and the process is repeated. Else if
10 the end of the reference tag table has not been reached,
another test is done to compare the selected tags of the
reference tag table and of the XML stream (box 230). If the
selected tags are different, the next tag of the reference
tag table is read (box 235) and the process is branched to
15 box 215 to determine whether or not the end of the reference
tag table has been reached.

If the tag entry in the reference tag table matches the
tag of the XML stream being processed , a new test is done
to determine if there are attributes associated to the
20 selected tag of the reference tag table (box 240). If there
is at least one attribute associated to this tag, the XML
parser gets the names of the attributes in the reference tag
table (box 245) and in the XML stream to determine if they
are the same (box 250). If the attribute names are the same,
25 the XML parser gets the attribute values (box 255), else the
process is branched to box 235 to select the next tag of the
reference tag table. Then, the XML parser checks if there is
a function name associated to the selected tag of the refer-
ence tag table (box 260). If there is no function name
30 associated to this tag, normal parsing continues (box 220),
the next tag of the XML stream is selected (box 225) and the
process is repeated. Else if there is a function name
associated to this tag, the XML parser gets the name of the

11

function (box 265) and executes the function with the
associated attribute values as parameters (box 270). Then,
the next tag of the XML stream is selected (box 225) and the
process is repeated.

5   If there is no attribute name associated to the
selected tag of the reference tag table (box 240), the XML
parser checks if there is a function name associated to the
selected tag of the reference tag table (box 260). If there
is no function name associated to this tag, normal parsing
10   continues (box 220), the next tag of the XML stream is
selected (box 225) and the process is repeated. Else if
there is a function name associated to the selected tag of
the reference tag table, the XML parser gets the name of the
function (box 265) and executes the function without attrib-
15   ute value i.e., without any parameter (box 270). The next
tag of the XML stream is then selected (box 225) and the
process is repeated until the XML parser reaches the end of
the XML stream.

        To illustrate the parser of the invention let consider
20   the following example of an XML document, the corresponding
optimized XML document and the reference tag table that is
associated to the corresponding DTD, consisting of the first
four rows of the reference tag table illustrated on figure
1. The aim of this illustration is to show how an XML
25   document may be optimized to reduce transmitting bandwidth
and parsing processing time, and how the optimized XML
parser handles it. The XML document (300) of the example
consists in a first image (305) that is duplicated to be
compared with two other images (310 and 320), each having a
30   title (315 and 325, respectively), as illustrated on figure
3.

12

In this example without usage of this invention, the XML document would be :

```
    <photo_compare>
      <first_compare>
5       <image>
          .................... image_reference data ...................
        </image>
        <title>Deutsche Bank</title>
        <image>
10        .................... image_1 data ...................
        </image>
      </first_compare>
      <second_compare>
        <image>
15        .................... image_reference data ...................
        </image>
        <title>Chateau Online</title>
        <image>
          .................... image_2 data ...................
20      </image>
      </second_compare>
    </photo_compare>
```

and the corresponding XML document, optimized according to the invention, is :

```
25  <photo_compare>
      <first_compare>
        <image save=image_reference skip=1500>
          .................... image_reference data ...................
        </image>
30      <title>Deutsche Bank</title>
        <image skip=1500>
          .................... image_1 data ...................
        </image>
      </first_compare>
35    <second_compare>
        <image include=image_reference>
```

```
        </image>
        <title>Chateau Online</title>
        <image skip=1500>
             .................... image_2 data ....................
  5     </image>
     </second_compare>
  </photo_compare>
```

In this example, the reference tag table associated to
the DTD of the XML document comprises one particular tag,
10 *image*, having three possible attributes, *save*, *skip* and
include. Each of the allowed combinations determines one of
the four functions, *SAVE_SKIP*, *SAVE*, *SKIP* and *INCL*, that may
be interpreted by the XML parser,

●   *SAVE_SKIP* function comprises two parameters, a name
15 i.e., a string of characters, and a length, i.e. a number.
SAVE_SKIP function stores the data that follows the tag in
the memory in association with the label given as the name
parameter and instructs the tag lookup function of the XML
parser to skip over the long tag-free data object that
20 follows , which results in decreased processing time. The
amount of data to be skipped is the second parameter of the
function and may be expressed e.g., as KBytes.

●   *SAVE* function has a name as parameter. *SAVE* function
stores the data following the tag in the memory, in associa-
25 tion with the label given as the parameter.

●   *SKIP* function allows the parser to skip the specified
length of data to decrease processing time. The parameter of
SKIP function is a data size e.g., a number of KBytes.

●   *INCL* function is used to insert data in the XML
30 document representation. Its parameter is a name to which
data is associated in the memory, the association being done
with *SAVE* or *SAVE_SKIP* function.

14

These functions are given as examples and the invention is not limited to those.

Therefore, when the XML parser of the invention receives the optimized XML stream, it analyzes the tags according to the algorithm described above by reference to figure 2. After having looked up and read the first tag, *photo_compare*, of the XML stream (box 200 of figure 2), the XML parser checks that it is not the end of the stream (box 205) and gets the first tag of the reference tag table consisting in the first four rows of the table of figure 1, (box 210). Since this tag is not the last one of the reference tag table (box 215) and it does not match with the selected tag of the XML stream (box 230), the next tag of the reference tag table is looked up (box 235) and the process is repeated. Since the first tag *photo_compare* does not belong to the corresponding reference tag table, normal parsing continues (box 220) and the next tag of the XML stream is get (box 225). Likewise, normal parsing is performed on the *first_compare* tag. The next tag, *image*, is then read and identified for special parsing.

Again, the first tag of the reference tag table is selected (box 210) to be compared with the selected tag of the XML stream (box 230). Since the tags are the same and there are attributes (box 240), the XML parser gets the attribute names from the reference tag table (box 245) and checks the XML stream to compare attribute names of the reference tag table and of the XML stream (box 250). Since attribute names match, the XML parser gets the attribute values (box 255), checks the presence of an associated function (box 260) and gets its name (box 265). The associated function is *SAVE_SKIP*. Thus, during function execution (box 270), the XML parser memorizes the image that follows

15

with the label *reference*, without analyzing the 1,500 KBytes
of data, *reference* and 1,500 KBytes being the parameters of
the function called by the tag *image* with the attributes
save and *skip*. Then, the next tag of the XML stream is read
5   and a normal processing continues until next tag *image* is
found.

Once again, the first tag of the reference tag table is
selected (box 210) to be compared with the tag being
processed in the XML stream (box 230). Since the tags are
10   the same and there are the attributes *save* and *skip* (box
240), the XML parser gets the attribute names from the
reference tag table (box 245) and checks the XML stream to
compare attribute names of the reference tag table and of
the XML stream (box 250). Since attribute names do not
15   match, the XML parser looks for the next tag in the refer-
ence tag table (box 235) and verifies that the end of the
reference tag table has not been reached (box 215). Since
the tags are the same (box 230) and there is the attribute
save (box 240), the XML parser gets the attribute name from
20   the reference tag table (box 245) and checks the XML stream
to compare attribute names of the reference tag table and of
the XML stream (box 250). Since attribute names do not
match, the XML parser looks for the next tag in the refer-
ence tag table (box 235) and verifies that the end of the
25   reference tag table has not been reached (box 215). The tags
being the same (box 230) and being in presence of the
attribute *skip* (box 240), the XML parser gets the attribute
name from the reference tag table (box 245) and checks the
XML stream to compare attribute names of the reference tag
30   table and of the XML stream (box 250). Since attribute names
match, the XML parser gets the attribute value from the XML
stream (box 255), checks the presence of an associated

16

function (box 260) and gets its name (box 265). The associated function is *SKIP*. Thus, during function execution (box 270), the XML parser does not analyze the following 1,500 KBytes of data i.e., the parser does not search any
5   tag in this stream portion, 1,500 KBytes being the parameter of the function called by tag *image* with the attribute *skip*. Then, the next tag of the XML stream is read and normal processing continues until the next tag *image* (having *include* as attribute) is found. The process is repeated
10 until the end of the stream.

At the end of the process, the optimized XML parser has built an internal tree representation of the optimized XML stream that may be used to provide data to a software application.

15   It is to be noticed that the size of the optimized XML stream is smaller than the one of the standard XML stream since redundant large data objects are not repeated. In the given example, the volume of data associated to large objects to be transmitted is reduced by twenty-five percent.
20 Also, the parsing processing time is reduced since the parser's tag lookup function saves useless search for tags in the remaining three large objects, which are known to be tag-less anyway. Finally, it is to be noted that depending upon the functions associated to the tags in the reference
25 tag table and the implementation of these functions, the optimized XML stream may be fully interpreted or not by a standard XML parser. For example, when considering *SKIP_PARSING* function, it allows to decrease the processing time when used within an optimized XML stream analyzed by a
30 parser handling reference tag table. When the optimized XML stream is analyzed by a standard parser it does not change the resulting internal parser representation, the only

17

difference lies in the processing time. However, when an optimized XML stream comprising functions like *PASTE* or *IMPORT* is analyzed by a standard parser, it leads to misinterpretation of the XML stream. In such case, the data is

5   not re-inserted at its place, which results in an incomplete interpretation of the document on the parser end.

As described above, the reference tag table is preferably associated to the Document Type Definition (DTD) of the XML document. Therefore, the reference tag table may be

10   transmitted with the associated DTD and the XML stream or may be stored by the XML parser. If the reference tag table is stored by the XML parser with the DTD, a dedicated tag of the XML stream allows the XML parser to establish the link between the XML stream and one of the stored DTD and refer-

15   ence tag table.

Naturally, in order to satisfy local and specific requirements, a person skilled in the art may apply to the solution described above many modifications and alterations all of which, however, are included within the scope of

20   protection of the invention as defined by the following claims.

18

Claims:

1. A method for optimizing tag based protocol stream parsing, using a reference tag table comprising at least one tag and a corresponding function name, said method compris-
5   ing, each time a tag is read from said tag based protocol stream, the steps of:

    - comparing said read tag and the tags of said reference tag table and,

    - if said read tag belongs to said reference tag table,
10   determining if a function name is associated to said tag belonging to said reference tag table and,

    - if a function name is associated to said tag belonging to said reference tag table, executing the function corre- sponding to said function name associated to said tag
15   belonging to said reference tag table.

2. The method of claim 1 wherein said reference tag table further comprises at least one attribute name associated to said at least one tag, the function corresponding to said function name associated to said tag belonging to said
20   reference tag table being executed only if said attribute name is associated to said read tag in said tag based proto- col stream.

3. The method of claim 2 wherein said attribute name associated to said read tag is associated to a value, said
25   value being used as a parameter by the function correspond- ing to said function name associated to said tag belonging to said reference tag table.

19

4.   The method of either claim 2 or claim 3 wherein a same
tag is associated to several set of attributes in said
reference tag table.

5.   The method of any one of claims 1 to 4 wherein the
5   function corresponding to said function name associated to
said tag belonging to said reference tag table comprises a
skip function that allows the parser to skip a determined
amount of data following said read tag, the amount of data
being either predetermined or being determined by an attrib-
10   ute value.

6.   The method of any one of claims 1 to 5 wherein the
function corresponding to said function name associated to
said tag belonging to said reference tag table comprises a
save function that associates, in memory, a determined
15   amount of data following said read tag to a predetermined
label or to a label being given as an attribute value.

7.   The method of any one of the previous claims wherein
said tag based protocol stream is an XML stream.

8.   The method of claim 7 wherein said reference tag table
20   is associated to the Document Type Definition defined in
said XML stream.

9.   An apparatus comprising means adapted for carrying out
the method according to any one of the claims 2 to 8.

20

10. A computer-like readable medium comprising instructions for carrying out the method according to any one of the claims 2 to 8.

100

	Tag	Attribute	Function
110-1	image	save, skip	SAVE_SKIP
110-2	image	save	SAVE
110-3	image	skip	SKIP
110-4	image	include	INCL
110-5	text	save, skip, style	S_S_S
110-6	text	save, skip	SAVE_SKIP
	....	....	....
110-I	tag(i)	attribute(i)	function(i)
	....	....	....
110-n	tag(n)	attribute(n)	function(n)

105-1   105-2   105-3

Figure 1

300

305

310

Deutsche Bank
315

320

Chateau Online
325
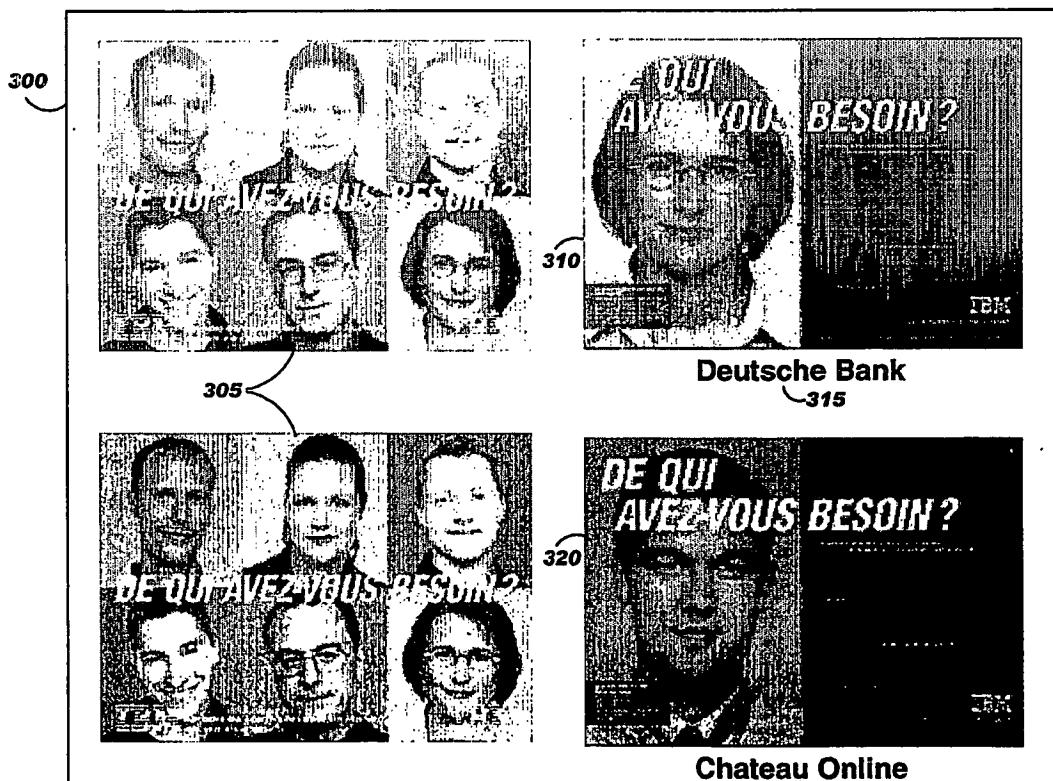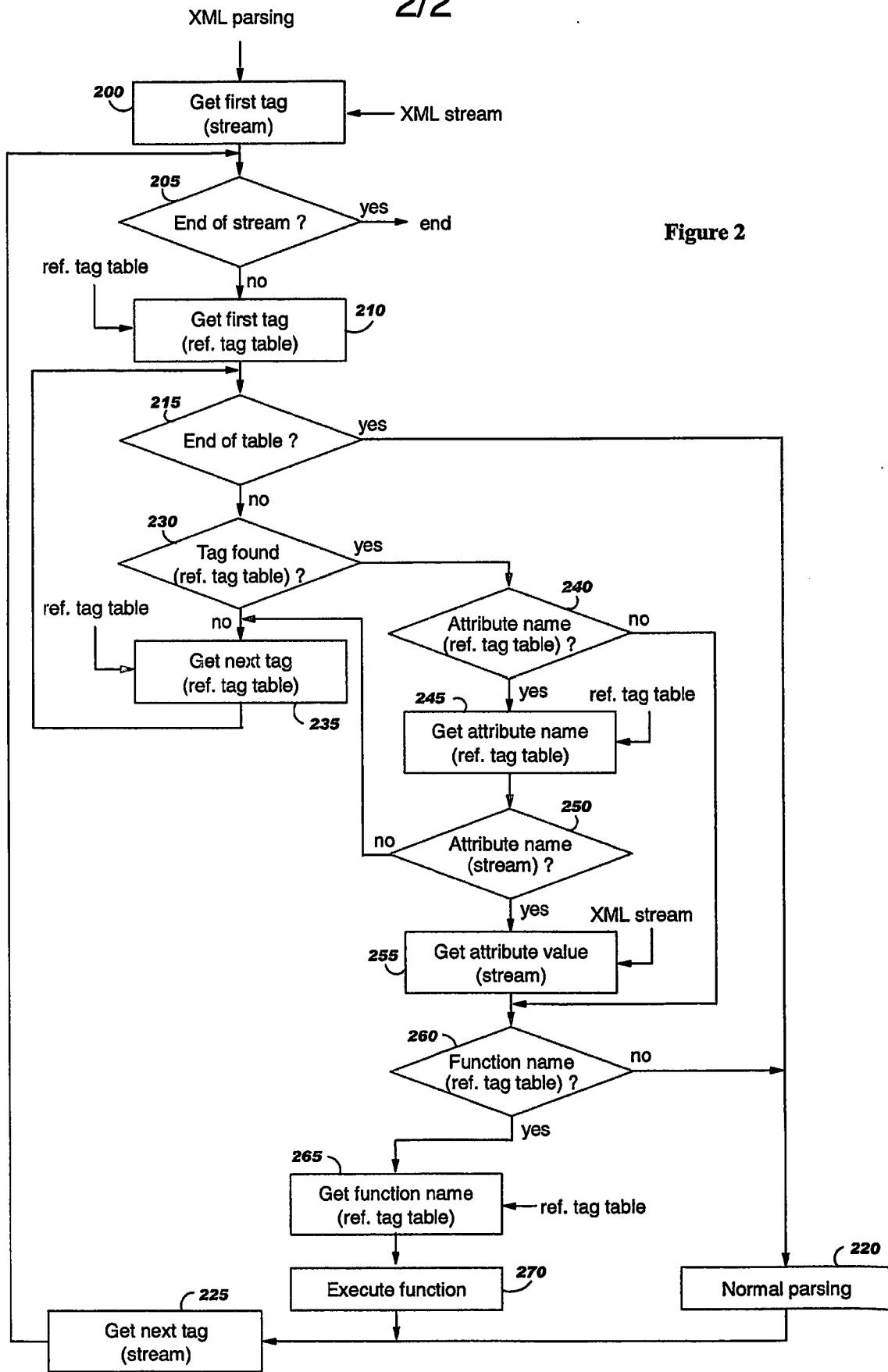
Figure 3

Figure 2

INTERNATIONAL SEARCH REPORT

Inter	Application No
	PCT/EP2004/002398

A. CLASSIFICATION OF SUBJECT MATTER
IPC 7   G06F17/22   G06F17/27

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
IPC 7   G06F

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)

EPO-Internal, WPI Data

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category°	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	FR 2 826 753 A (CANON KK) 3 January 2003 (2003-01-03) page 4, line 17 - line 25 -----	1-4,9,10
X	US 6 480 865 B1 (LEE SUSAN B ET AL) 12 November 2002 (2002-11-12) cited in the application column 4, line 67 - column 5, line 8 -----	1-4,9,10

☐ Further documents are listed in the continuation of box C.

☒ Patent family members are listed in annex.

° Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
31 August 2004	10/09/2004

Name and mailing address of the ISA	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Tx. 31 651 epo nl, Fax: (+31-70) 340-3016	Triest, J

Form PCT/ISA/210 (second sheet) (January 2004)

Patent document cited in search report		Publication date	Patent family member(s)		Publication date
FR 2826753	A	03-01-2003	FR	2826753 A1	03-01-2003
			JP	2003132043 A	09-05-2003
			US	2003069907 A1	10-04-2003
US 6480865	B1	12-11-2002	NONE		